Supplementary Material
“Do Not Marginalize Mechanisms, Rather Consolidate!”

A Evaluation of Partitioned SCM

A partitioned SCM $\mathcal{M}_{\mathcal{A}}$ consists of several sub SCM $\mathcal{M}_{\mathbf{A}}$, that, in sum, cover all variables and structural equations of an initial SCM $\mathcal{M}$. Thus, evaluation of a partitioned SCM yields the same set of values $\mathbf{v} \in \mathbf{V}$ as the original $\mathcal{M}$. Similar to the evaluation of structural equation in the initial $\mathcal{M}$, sub SCM need to be evaluated in a specific order to guarantee all $\mathbf{u} \in \mathcal{M}'_{\mathbf{U}}$ exist. As such, sub SCM can be considered multivariate variables that establish another high-level DAG. The evaluation order is determined via the relation $\mathrm{R}_{\mathbf{X}}$ as defined in Sec. 3.1 and depends on the graph partition $\mathcal{A}$ and the order of $\mathbf{X}$ imposed by the the initial SCM.

Algorithm 1 Evaluation of partitioned SCM

1: **procedure** PARTITIONEDSCMEVAL($\mathcal{M}_{\mathcal{A}}, \mathbf{u}, \mathbf{I}$)
2:     $\mathbf{x} \leftarrow \mathbf{u}$ ▷ $\mathbf{x}$ will gradually collect all values $\mathbf{x} \in \mathbf{X}$ of $\mathcal{M}$
3:     **for A in** sort($\mathcal{A}, \mathrm{R}_{\mathbf{X}}$) **do** ▷ Sort Clusters by strict partial order imposed by $\mathcal{M}$
4:         $\mathcal{M}'_{\mathbf{A}} \leftarrow \mathcal{M}'_{\mathbf{A}'} \in \mathcal{M}_{\mathcal{A}}$ `where` $\mathbf{A}' = \mathbf{A}$
5:         $\mathbf{u}' \leftarrow \{x_i \in \mathbf{x} \mid \mathbf{X}_i \in \mathcal{M}'_{\mathbf{U}}\}$
6:         $\mathbf{I}' \leftarrow \psi_{\mathbf{A}}(\mathbf{I})$
7:         $\mathbf{v} = \mathcal{M}'^{\mathbf{I}'}_{\mathbf{A}}(\mathbf{u}')$
8:         $\mathbf{x} = \mathbf{x} \cup \mathbf{v}$
9:     **end for**
10:     $\mathbf{v} = \{x_i \in \mathbf{x} \mid \mathbf{X}_i \in \mathcal{M}'_{\mathbf{V}}\}$ ▷ Filter all **u** ∈ **U** to get **v** ∈ **V**
11:     **return v**
12: **end procedure**

Algorithm 1 shows the evaluation of partitioned SCM, where $\mathcal{M}_{\mathcal{A}}$ is the partitioned SCM we want to evaluate, $\mathbf{u}$ are the values of exogenous variables to the initial model $\mathcal{M}$ and $\mathbf{I}$ is the set of applied interventions. The outcomes of sub SCM that are not related via $\mathrm{R}_{\mathbf{X}}$ are invariant to the evaluation order among each other. Even though $\mathrm{R}_{\mathbf{X}}$ defines the ordering of sub SCM only up to some partial order, sort($\mathcal{A}, \mathrm{R}_{\mathbf{X}}$) can pick any total ordering that is valid with $\mathrm{R}_{\mathbf{X}}$.

Proof 1 (Consistency of Partitioned SCM Evaluation) *Evaluations of $\mathcal{M}'_{\mathbf{A}}$ every, in step 7, compute all variables $\mathbf{V}_i \in \mathbf{A}$ by evaluating f_i of the original SCM, yielding the same values as the evaluation of $\mathbf{A}$ in $\mathcal{M}$. Therefore $\mathrm{P}_{\mathcal{M}'_{\mathbf{A}}} = \mathrm{P}_{\mathcal{M}_{\mathbf{A}}}$. By Def. 4 every variable $V \in \mathbf{V}$ is contained within some sub SCM $\mathcal{M}'_{\mathbf{A}}$. The evaluation of* `PartitionedSCMEval` *is complete, in the sense that all $\mathbf{V} = \bigcup \mathcal{A} = \bigcup_{\mathbf{A} \in \mathcal{A}} \mathbf{A}$ are evaluated, as the evaluation of all $\mathcal{M}'_{\mathbf{A}} \in \mathcal{M}_{\mathcal{A}}$ is guaranteed by iterating over all $\mathbf{A}$ in step 2. Finally $\mathrm{P}_{\mathcal{M}'_{\mathcal{A}}} = \bigcup_{\mathbf{A} \in \mathcal{A}} \mathrm{P}_{\mathcal{M}'_{\mathbf{A}}} = \bigcup_{\mathbf{A} \in \mathcal{A}} \mathrm{P}_{\mathcal{M}_{\mathbf{A}}} = \mathrm{P}_{\mathcal{M}_{\mathbf{V}}}$.* ∎

B Complexity reduction in function composition

Reduction of encoding length might vary depending on the type and structure of the equations under consideration. No compression of structural equation is gained when the system of consolidated equations is already minimal. Compression of equation to an identity function is showcased in the following.

B.1 Compression of chained inverses

Reduction to constant complexity for the unintervened system is reached in the case of $f_B = f_A^{-1}$. Consider the equation chain of $X \to A \to B$ with A getting marginalized. Immediately $f'_B := f_B \circ f_A = f_A^{-1} \circ f_A = \mathrm{Id}$ follows. Therefore, $B := X$, which is a single assignment of the value(s) of X into B. Remaining complexity within the consolidated function is then only due to conditional branching in cases of $do(A = a), do(B = b) \in \mathbf{I}$.

B.2 Matrix composition is not sufficient for compressing equations

The operation of matrix multiplication, as a way of expressing composition of linear functions, stays within the class of matrices. Matrix multiplication, therefore, serves as a possible candidate to be considered when consolidating equations and reducing the encoding length of a linear structural systems. When written down an a 'high-level' view, matrices can expressed in terms of single variables $A, B \in \mathbb{R}^{M\times N}$ and matrix multiplication $\times : \mathbb{R}^{M\times N} \times \mathbb{R}^{N\times O} \to \mathbb{R}^{M\times O}$. Assuming equations $f_Y := A \times X$ and $f_Z := B \times X$, we can reduce the length of the composed equation $f'_Z := A \times B \times X$ by multiply the matrices A and B together, $f_i = C \times X$ with $C = A \times B$. While we effectively reduced the number of high-level symbols written in the equation, we are hiding computational complexity in the structure of the matrix C. The following simple counterexample demonstrates a situation where the size, as well as, the number of non-zero entries even increases:

$$\overset{C}{\begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}} = \overset{A}{\begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}} \times \overset{B}{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}}$$

Thus, proving that pure matrix multiplication, is not suitable to keep, or even minimize, the size of composed function representations.

B.3 Compression over Finite Discrete Domains

Consolidation may reduce the number of variables within a graph, but burdens the remaining equations with the complexity of the consolidated variables. Without the need to explicitly compute values of consolidated variables, we might leverage cancellation effects to simplify equations, as outlined in the main paper. In terms of compression, no guarantees can be given in the general case. However, we will now show, that the often considered case of chained maps between finite discrete domains simplifies or at least preserves complexity.

The cardinality of the image of a deterministic function $f : \mathcal{X} \to \mathcal{Y}$ between two finite discrete sets $\mathcal{X}, \mathcal{Y}$ is bounded by the cardinality of its domain: $|\operatorname{Img}(f)| \leq |\operatorname{Dom}(f)| \leq |\mathcal{X}|$, where $\operatorname{Img}(f)$ is the image and $\operatorname{Dom}(f)$ the domain of f. In particular, the strict inequality $|\operatorname{Img}(f)| < |\operatorname{Dom}(f)|$ holds for all non-injective maps. Function composition may further reduce the 'effective' domain $\operatorname{Dom}_{\text{effective}}(f)$ of a function, by only considering values of the image of the previous map as inputs to the next function. In contrast considering to all possible values of $\mathcal{X}$ in the case of the non-composed map, the image of the previous function may only be a subset of $\mathcal{X}$. Therefore, $f_2 \circ f_1 \Rightarrow |\operatorname{Img}_{\text{effective}}(f_2)| \leq |\operatorname{Dom}_{\text{effective}}(f_2)| = |\operatorname{Img}(f_1)| \leq |\operatorname{Dom}(f_1)|$. In particular, the effective image of a composition chain $f_n \circ \cdots \circ f_1$ is bounded by the function with the smallest image: $|\operatorname{Img}_{\text{effective}}(f_n \circ \cdots \circ f_1)| \leq \min |\operatorname{Img}(f_i)|$. Thus, equation chains over finite discrete domains strictly preserve or reduce the effective size of the image, allowing for a possibly simpler combined representation in comparison to representing the functions individually.

C Reparameterization of non-deterministic structural equations.

Consolidation of structural equations might lead to duplication of non-deterministic terms within consolidated systems. For example when consolidating fork structures (compare to Sec. 4.1). Without further precautions, different values might be sampled from the duplicated non-deterministic equations. An example where consolidating a variable B with a non-deterministic equation f_B (indicated by a squiggly line) leads to inconsistent behaviour is shown in 5. In $\mathcal{M}_1$, C and D both copy on the value of B. Therefore, $c = d$ yields always. $\mathcal{M}_{1'}$ shows a graph where B is consolidated from $\mathcal{M}_1$. As a result the non-deterministic equation f_B is duplicated into the equations of C and D, such that $f_C := \operatorname{Bern}(A)$ and $f_D := \operatorname{Bern}(A)$. Within the consolidated model $\mathcal{M}_{1'}$ different values might be sampled from the different noise terms $\operatorname{Bern}(A)$ in f_C and f_D. Consequently $c \neq d$ might occur in $\mathcal{M}_{1'}$. To obtain consistent behaviour with the initial $\mathcal{M}_1$, we need to ensure agreement about the value of $\operatorname{Bern}(A)$ across all instances of the duplicated equation. To do so, we reparameterize $\mathcal{M}_1$ and explicitly store a fixed value, sampled from $\operatorname{Bern}(A)$, into a new exogenous variable R. The equation f_B is then reparameterized into a deterministic structural equation taking the variable R as an additional argument, resulting in $\mathcal{M}_2$. When consolidating B within $\mathcal{M}_2$, all instances of f_B now yield the same value, as the noise term is fixed via R and finally $\mathrm{P}_{\mathcal{M}'_2} = \mathrm{P}_{\mathcal{M}_1}$.

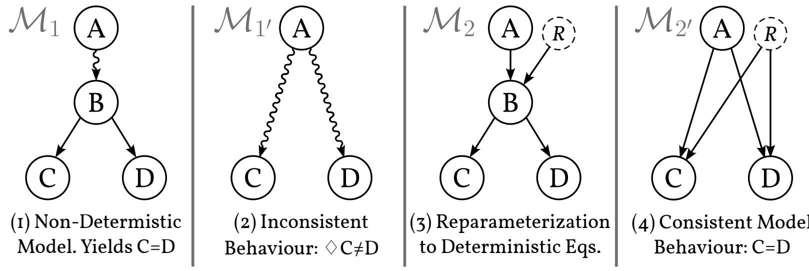

Figure 5: **Reparameterization of non-deterministic models.** The SCM $\mathcal{M}_1$ contains a non-deterministic equation $B := \text{Bern}(A)$ (marked with a squiggly line). With $C := B$ and $D := B$, $\mathcal{M}_1$ always yields $C = D$. Simply consolidating (or marginalizing) B creates a model $\mathcal{M}_{1'}$ with $C := \text{Bern}(A)$ and $D := \text{Bern}(A)$, such that possibly $C \neq D$. Reparameterizing f_B by introducing an exogenous random variable $R := \mathcal{U}(0, 1)$ and $B := A < R$, yields the SCM $\mathcal{M}_2$ with only deterministic equations. Consolidating (or marginalizing) B in $\mathcal{M}_2$ leads to $\mathcal{M}_{2'}$ where $C := A < R$ and $D := A < R$, thus always $C = D$.

D Consolidation Examples

In this section we show further detailed applications of consolidation. Section D.1 presents the worked out consolidation of the dominoes motivating example of the paper, with regard to generalizing abilities of consolidates models. Section D.2 considers consolidation of the classical firing squad example. In contrast to the other examples, we focus on consolidating graphs with multiple edges in the causal graph. Lastly we provide the causal graph and structural equations of the game agent policy discussed in the main paper, in Section D.3.

D.1 Motivating Example: Dominoes

While we applied consolidation to a particular SCMs in the main paper, we will discuss the motivating example with focus on obtaining representations that cover generalize over populations of SCM. We demonstrate this on the particular example of a rows of dominoes, as a simple SCM with highly homogenous structure. Regardless of whether the SCM is obtained by using methods for direct identification of causal graphs from image data, as presented by Brehmer et al. [2022], or abstracting physical simulation using τ-abstractions [Beckers and Halpern, 2019]; we assume to be provided with a binary representation of the domino stones. The state of every domino S_i indicates whether it is standing up or getting pushed over. In this case, the structural equations for all dominoes are the same: $f_i := S_{i-1}$. As a result tipping over the first stone in a row will lead to all stones falling. Also, we are only interested in the final outcome of the chain. That is, whether the last stone will fall or not ($\mathbf{E} = \{S_n\}$). Again, we use consolidation to collapse the structural equations in the unintervened case: $S_n := f_n \circ \cdots \circ f_1 := S_1$. We consider a single active allowed intervention of holding up any of the dominoes or tipping it over, $\mathcal{I} = \{do(S_i = 0), do(S_i = 1)\}$. Upon evaluation, the unconsolidated model needs to check for every domino if it is being intervened or not, requiring n conditional branches. Using the fact that perfect interventions 'overwrite' the variable state for the following dominoes, we introduce a first order quantifier that handles all intervention in a unified way. Finally, by combining the formulas of the intervened and unintervened case, we find the following simple equation:

$$S_n := \begin{cases} x_i & \text{if } \exists\, do(S_i = x_i) \in \mathbf{I} \\ S_1 & \text{else} \end{cases}$$

The resulting equation no longer has a notion of the actual number of dominoes and, in fact, it is invariant to it. We realise that introducing the first-order for-all $\forall$ and exists $\exists$ quantifiers allows for a unified representation of arbitrary chains of dominoes. Similar observations are discussed in Peters and Halpern [2021] and Halpern and Peters [2022] which introduce generalized SEM (GSEM). As intermediate the equations are no longer computed explicitly, the structural equations of consolidated models for different row lengths only differ in the set of allowed interventions $\mathcal{I}$. That is, for a row of three domino stones $\mathcal{I} = \{do(V_1 = v_1), do(V_2 = v_1), do(V_3 = v_1)\}$, while for four stones the additional $do(V_4 = v_1)$ is defined. As set out in the introduction of this paper, we consider consolidation as a tool for obtaining more interpretable SCM. Towards this end, consolidation might help us in detecting similar structures within an SCM. Doing so eases understanding of causal systems, as the user only has to understand the general mechanisms of a particular SCM once and is then able to apply the gained knowledge to all newly appearing SCM of the same type.

D.2 Firing Squad Example

While the dominoes and tool wear examples where mainly considering the consolidation of sequential structures, we want to briefly demonstrate the consolidation of structural equations that are arranged in a parallel fashion. We consider a variation of the well known firing squad example [Hopkins and Pearl, 2007] with a variable number N of rifleman. A commander (C) gives orders to rifleman ($R_i, i \in \{1 \dots N\}$), which shoot accurately and the prisoner (P) dies. For the sequential stacking of equations we found that interventions exert an 'overwriting' effect. That is, every intervention fixes the value of a variable, making the unfolding of the following equations independent of all previous computations. To yield a similar effect for parallel equations we need to block *all* paths between the cause and effect. In this scenario, this can easily be expressed by using an all-quantifier. When consolidating the SCM, we consider only the captain C and prisoner P, $\mathbf{E} = \{C, P\}$, while allowing for any combination of interventions that prevent the rifleman from shooting $\mathcal{I} = \mathcal{P}(\{do(R_i = 0)\}_{i \in \{1 \dots N\}})$. After consolidation, we obtain the following equation:

$$\text{P} := \begin{cases} \text{lives} & \text{if } C = 0 \vee (\forall S_i.\, do(S_i = 0) \in \mathbf{I}) \\ \text{dies} & \text{else} \end{cases}$$

As with the dominoes example, we are again in a situation where the consolidated equation intuitively summarizes the effects of individual: "The prisoner lives if the captain does not give orders, or if all riflemen are prevented from shooting".

D.3 Revealing Agent Policy: Causal Graph and Equations

In this section we explicitly list the structural equations representing observed interactions between a platformer environment and a possible rule based agent. The resulting causal graph is shown in Fig.6 at the end of the appendix. Except for the parentless variables 'coin_reward', 'powerup_reward', 'enemy_reward', 'flag_reward', 'player_position', 'position_coin', 'position_powerup', 'position_enemy', 'position_flag' and 'target_flag', which are exogenous and determined by the environment, all variables are considered endogenous:

$$
\begin{aligned}
&\text{player\_position}, \text{position\_coin}, \text{position\_powerup}, \text{position\_enemy}, \text{position\_flag} \in [0..1]^2 \\
&\text{coin\_reward} := 3; \text{powerup\_reward} := 1; \text{enemy\_reward} := 9; \text{flag\_reward} := 2 \\
&\text{With } X \text{ in } \{\text{coin}, \text{powerup}, \text{enemy}, \text{flag}\}: \\
&\quad \text{distance\_}X := \|\text{position\_}X - \text{player\_position\_}X\|_2 \\
&\quad \text{near\_}X := \text{distance\_}X < 3.0 \\
&\quad \text{targeting\_cost\_}X := 1.0 + 0.5 \times \text{distance\_}X \\
&\text{target\_coin} := \text{targeting\_cost\_coin} < \text{enemy\_reward} \\
&\text{target\_powerup} := \text{targeting\_cost\_powerup} < \text{powerup\_reward} \\
&\text{target\_enemy} := \text{targeting\_cost\_enemy} < \text{enemy\_reward} \wedge \text{powered\_up} \\
&\text{target\_flag} := \text{True} \\
&\text{powered\_up} := \text{target\_powerup} \\
\\
&\text{towards\_coin} := \text{target\_coin} \wedge \text{coin\_reward} > \max(\{X\text{\_reward}|\text{target\_}X\}_{X \in \{\text{powerup},\text{enemy},\text{flag}\}}) \\
&\text{towards\_powerup} := \text{target\_powerup} \wedge \text{powerup\_reward} > \max(\{X\text{\_reward}|\text{target\_}X\}_{X \in \{\text{coin},\text{enemy},\text{flag}\}}) \\
&\text{towards\_enemy} := \text{target\_enemy} \wedge \text{enemy\_reward} > \max(\{X\text{\_reward}|\text{target\_}X\}_{X \in \{\text{enemy},\text{powerup},\text{flag}\}}) \\
&\text{towards\_flag} := \text{target\_flag} \wedge \text{flag\_reward} > \max(\{X\text{\_reward}|\text{target\_}X\}_{X \in \{\text{coin},\text{powerup},\text{enemy}\}}) \\
&\text{jump} := \text{near\_enemy} \wedge \neg\text{powered\_up}
\end{aligned}
$$

$$
\text{planning\_sequence}_i := \begin{cases} \text{finished} & \text{if towards\_flag} \wedge (\text{flag} \in \bigcup_{j=1}^{i-1} \text{planning\_sequence\_}j) \\ \text{coin} & \text{if towards\_coin} \wedge (\text{coin} \notin \bigcup_{j=1}^{i-1} \text{planning\_sequence\_}j) \\ \text{powerup} & \text{if towards\_powerup} \wedge (\text{powerup} \notin \bigcup_{j=1}^{i-1} \text{planning\_sequence\_}j) \\ \text{enemy} & \text{if towards\_enemy} \wedge (\text{enemy} \notin \bigcup_{j=1}^{i-1} \text{planning\_sequence\_}j) \\ \text{flag} & \text{if towards\_flag} \wedge (\text{flag} \notin \bigcup_{j=1}^{i-1} \text{planning\_sequence\_}j) \\ \text{finished} & \text{else} \end{cases}
$$

$$
\begin{aligned}
\text{score} := 20 &- \text{time\_taken} \\
&+ \text{coin\_reward if coin} \in \text{planning\_sequence}_i \\
&+ \text{powerup\_reward if powerup} \in \text{planning\_sequence}_i \\
&+ \text{enemy\_reward if enemy} \in \text{planning\_sequence}_i \wedge \text{powerup} \in \text{planning\_sequence}_i \\
&+ \text{flag\_reward if flag} \in \text{planning\_sequence}_i
\end{aligned}
$$

E Mathematical symbols and notation

The following table contains mathematical functions and notation used throughout the paper.

Notation	Meaning
X; $\mathbf{X}$	A (set of) variable(s).
x; $\mathbf{x}$	Value(s) of X; $\mathbf{X}$.
$\mathbf{X}_i$	The i-th variable of $\mathbf{X}$.
$\mathbf{X}_\mathbf{S}$	The subset $\{\mathbf{X}_i : i \in \mathbf{S}\}$ of $\mathbf{X}$.
$\mathrm{P}_\mathbf{X}$	A probability distribution over variables $\mathbf{X}$.
$x \sim \mathrm{P}_X$	A value x sampled from a distribution over X.
$\mathcal{P}(\cdot)$	The power set.
$f \circ g$	Function composition, $(f \circ g)(x) = f(g(x))$.
$\prod_{X_i \in \mathbf{X}} \mathcal{X}_i$	N-ary Cartesian product over the domain of $\mathbf{X}$.
$\lVert\cdot\rVert_2$	l^2 vector norm.
$\mathcal{U}(a, b)$	Uniform Distribution.
$\mathcal{N}(\mu, \sigma^2)$	Normal Distribution.
$\mathrm{Bern}(p)$	Bernoulli distribution; Takes value 1 with probability p and 0 otherwise.
$\mathrm{P}_\mathcal{M}$	Probability distribution over the SCM $\mathcal{M}$.
$\mathrm{P}^\mathbf{I}_\mathcal{M}$	Probability distribution over the SCM $\mathcal{M}$ under intervention $\mathbf{I}$.
V_i	An endogenous variable of an SCM $\mathcal{M}$.
U_i	An exogenous variable of an SCM $\mathcal{M}$.
f_i	Structural equation of the variable X_i.

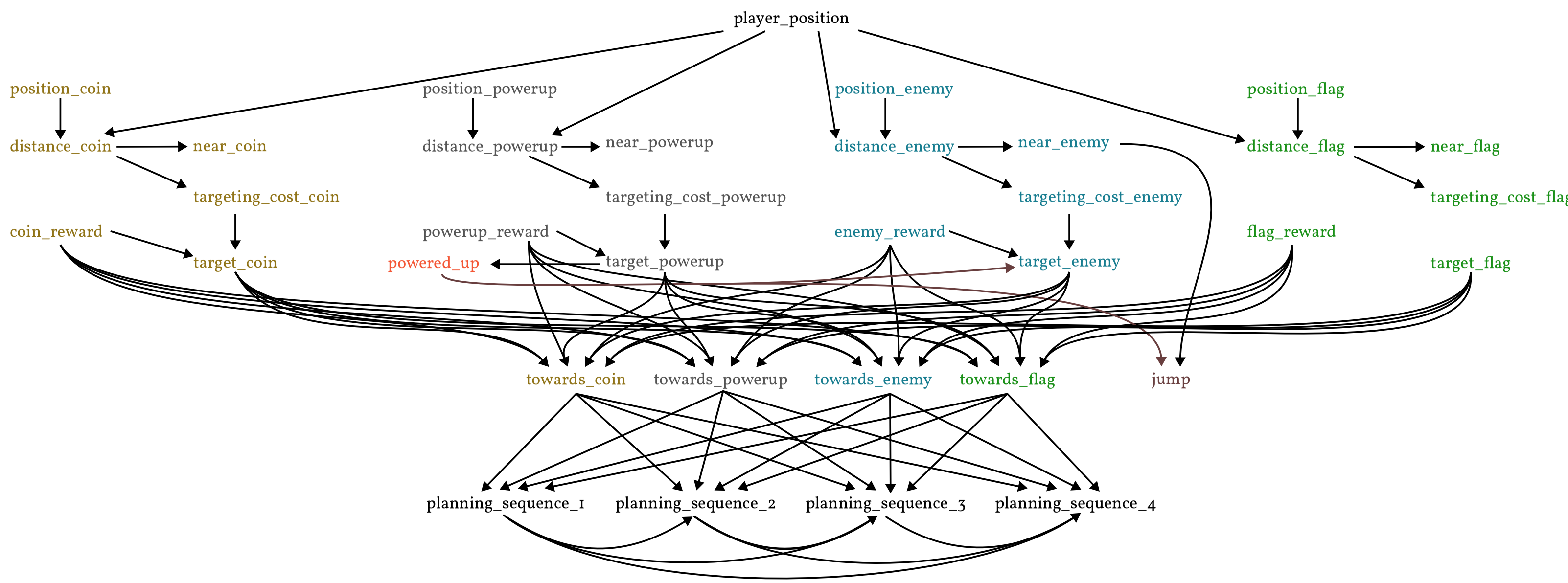

Figure 6: **Causal graph of an agent policy.** The causal graph of a greedy agent inside an platformer environment. The parentless variables are exogenous. Their value is determined via the game environment. The final ‘score’ variable is left out for clarity.